CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 Kiểu STRUCT
13.2 Kiểu UNION
13.3 Kiểu ENUM *(Enumerated)*
13.4 Định nghĩa kiểu bằng TYPEDEF
Bài tập cuối chương

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Struct (tạm dịch là cấu trúc) là một kiểu dữ liệu phức hợp được tạo từ các kiểu dữ liệu khác, các kiểu dữ liệu này được sử dụng khai báo cho các biến thành phần của biến kiểu struct.

```
struct tên_cấu_trúc
        {
                Khai báo các biến thành phần
        };
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

```
struct sinh_viên
        {
                char ma_so[10];
                char ho_ten[40];
                int tuoi;
                char dia_chi[80];
        };
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Cú pháp của một khai báo biến cấu trúc giống như khai báo biến bình thường:

struct tên_struct tên_biến;

Ví dụ:

struct sinh_vien sv1, sv2;

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Ví dụ:

```
struct sinh_viên
        {
                char ma_so[10];
                char ho_ten[40];
                int tuoi;
                char dia_chi[80];
        }       sv1, sv2;
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

10 byte	40 byte	2 byte	80 byte
ma_so	ho_ten	tuoi	dia_chi

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Ví dụ:

```
struct sinh_vien sv1 = { "4950897", "Tran van Vinh", 21,
"42 Truong Cong Dinh p.13 q.TB"};
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Để truy xuất một thành phần của biến cấu trúc, C có toán tử chấm "." để lấy từng thành phần.

Ví dụ:

strcpy (*sv1.ma_so*, 4950897");
strcpy (*sv1.ho_ten*, Tran van Dinh");
sv1.tuoi = 21;
strcpy (*sv1.dia_chi*, "42 Truong Cong Dinh p.13 q.TB");

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

C cho phép gán các cấu trúc cùng kiểu cho nhau qua tên biến cấu trúc thay vì phải gán từng thành phần cho nhau.
Ví dụ:

```
sv2 = sv1;
```

Ví dụ 14.8 (GT)

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Các thành phần của biến struct cũng là biến bình thường, nên ta có thể lấy địa chỉ của chúng, địa chỉ này là một hằng pointer trỏ đến thành phần tương ứng.
Ví dụ 14.9 (GT)

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.1 Khái niệm - Khai báo struct

Kiểu struct có thể được lấy kích thước tính theo byte nhờ toán tử sizeof, ví dụ:

```
sizeof (struct sinh_vien);
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.2 Mảng các struct

Cú pháp khai báo mảng các struct:

struct ten_cau_truc ten_mang [kich_thuoc];

Ví dụ:

```
struct sinh_vien sv[50];
strcpy (sv[0].ho_ten, “Dang thanh Tin”);
sv[0].tuoi = 28;
```

Ví dụ 14.12(SGT)

13.1 KIỂU STRUCT

13.1.3 Pointer tới một struct

Cú pháp khai báo biến pointer này như sau:

```
struct tên_cấu_trúc *tên_pointer;
```

Ví dụ :

```
struct sinh_vien a, *psv;
psv =&a;
```

hoặc

```
struct sinh_vien sv[20], *psv;
psv =sv;
```

13.1 KIỂU STRUCT

13.1.3 Pointer tới một struct

Việc truy xuất đến một thành phần của một cấu trúc thông qua một pointer được thực hiện bằng toán tử lấy thành phần của đối tượng của pointer, ký hiệu là -> (có thể gọi là toán tử mũi tên).

Ví dụ:

```
printf ("Ho ten sinh vien: %s \n", psv -> ho_ten);
```

hay

```
printf ("Ho ten sinh vien: %s \n", (*psv).ho_ten);
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.3 Pointer tới một struct

Ví dụ 7.16 (SGT)

C lại cho phép khai báo struct mà trong các thành phần của nó lại có các pointer chỉ đến một cấu trúc cùng kiểu.

Ví dụ:

```
struct node
        {
                char message[81];
                struct node *next;          };
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.4 Struct dạng field

C cho phép ta khai báo các thành phần của struct theo bit hoặc một nhóm bit. Một thành phần như vậy được gọi là một field (tạm dịch là vùng).

struct tên_cấu_trúc
{ **kiểu** tên_vùng 1: số_bit1;
kiểu tên_vùng 2:số_bit2;
...
} **tên_biến**;

Với **kiểu** chỉ có thể là unsigned, signed hoặc int

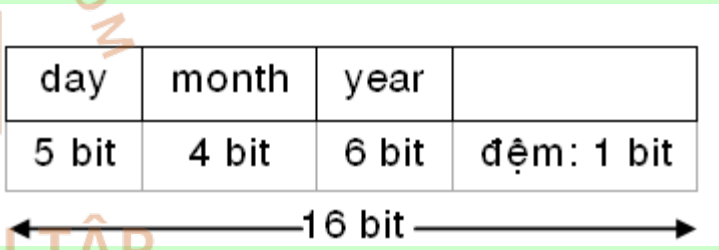
CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.4 Struct dạng field

Ví dụ:

```
struct date
        {
                unsigned day: 5;
        unsigned month: 4;
                unsigned year: 6;
        int: 0;
        } ngay;
```



CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.4 Struct dạng field

Chú ý:

-Mỗi vùng chỉ có thể dài tối đa 16 bit (một int) và được cấp chỗ trong một int, chứ không thể nằm trên hai int khác nhau được.

-Sự phân bố bit cho các field trong một int của struct (từ trái sang phải hay ngược lại), không phân biệt được.

-Mọi thao tác thực hiện trên biến kiểu field có liên quan đến địa chỉ đều không được thực hiện

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.4 Struct dạng field

Ví dụ:

Khi khai báo

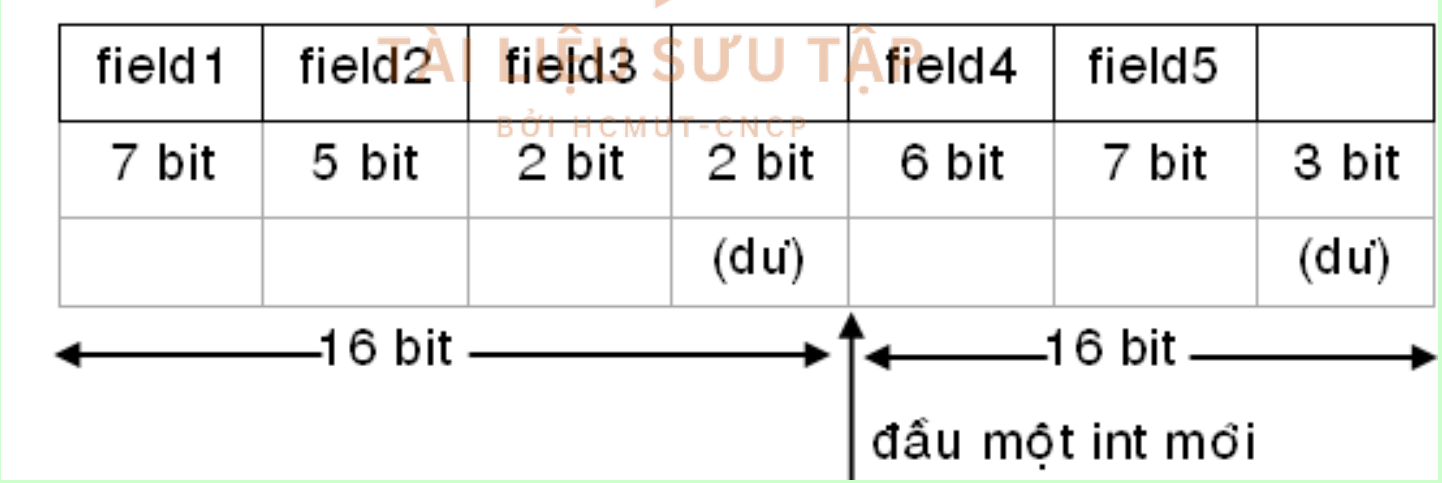
```
struct vi_du
        {       unsigned field1: 7;
                unsigned field2: 5;
                unsigned field3: 2;
                unsigned field4: 6;
                unsigned field5: 7;         } vd;
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.1 KIỂU STRUCT

13.1.4 Struct dạng field

Ví dụ:



CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.2 KIỂU UNION

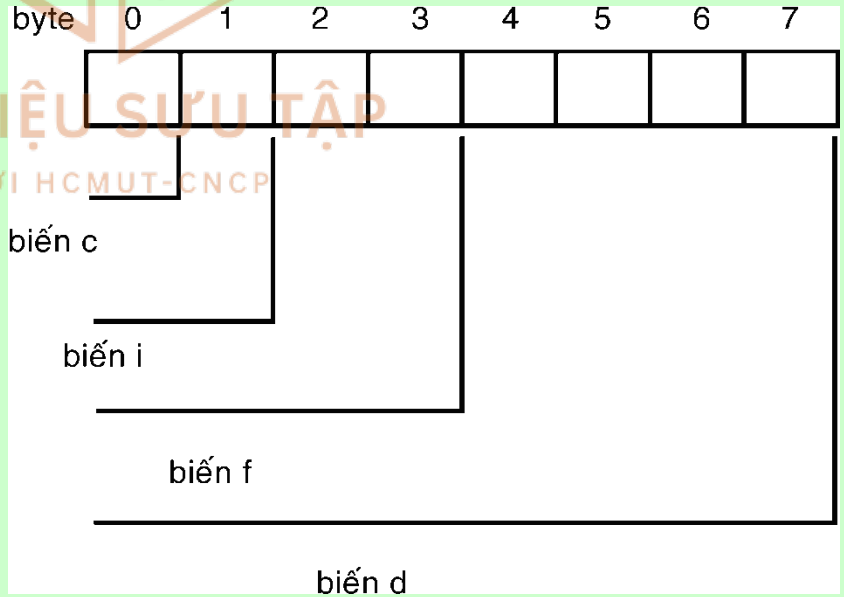
Trong ngôn ngữ C có kiểu dữ liệu union (tạm dịch là kiểu hợp nhất), đây là một kiểu dữ liệu đặc biệt mà nếu được khai báo thì ứng với một vùng nhớ, giá trị ở mỗi thời điểm khác nhau thì có thể có kiểu khác nhau tùy vào việc sử dụng biến thành phần trong nó.

13.2 KIỂU UNION

Ví dụ:

Có khai báo union như sau:

```
union thu
{
        char c;
        int i;
        float f;
        double d;
};
```



CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.2 KIỂU UNION

Khai báo biến kiểu union:

```
union tên_union
{
        khai_báo_biến_thành_phần
}       biến, biến [,...];
```

hoặc

```
union tên_union biến, biến [...];
```

13.2 KIỂU UNION

Ví dụ:

```
union thu
{       char c;
        int i;
        float f;
        double d;
} a, b;
```

hoặc

```
union thu a, b;
```

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.2 KIỂU UNION

Để truy xuất đến một biến thành phần của biến thuộc kiểu uinion, ta cũng dùng toán tử chấm ".".

Ví dụ:

```
union thu a;
a.c = 'a';
```

Ta có thể khai báo một biến pointer chỉ đến một biến kiểu union. **Ví dụ:**

```
union thu *pthu, a;
pthu = &a;
```

13.2 KIỂU UNION

Việc truy xuất đến một thành phần của union qua pointer cũng được thực hiện bằng toán tử mũi tên, để lấy thành phần của union đang được pointer chỉ đến.

Ví dụ:

pthu->c = 'A';

Ví dụ 7.30 (SGT)

Kiểu union có thể được lấy kích thước tính theo byte qua toán tử sizeof, ví dụ: sizeof (union thu);

CHƯƠNG 13
CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC VÀ KIỂU DỮ LIỆU TỰ ĐỊNH NGHĨA

13.3 KIỂU ENUM *(ENUMERATED)*

13.4 ĐỊNH NGHĨA KIỂU BẰNG TYPEDEF